Inaba Denki Sangyo Co., Ltd.
CHOCO TEI WATCHER mini-providers
User's Guide

Version 1.0.1

June 28, 2022

NOTE:

[Revision History]

Version	Date	Description
1.0.0	2020-11-30	First edition
1.0.1	2022-06-22	Fixed a bug in the GetErrorLog command. Fixed a bug in the completion message of the GetMovie command. Improved internal processing of @ERROR_CODE variable. Fixed a bug of @IMAGE variable immediately after mode switching.

[Compatible models]

Model	Version	Notes
IB-MCT001	--	

[Operation check model]

Model	Version	Notes
IB-MCT001	72n	

Contents

1. Introduction

This manual is a user's guide for providers that connect to CHOCO TEI WATCHER mini (Model: IB-MCT001) of Inaba Electric Industry Co., Ltd. Fig. 1-1 shows the overall configuration of this provider and the device. From now on, the provider is simply referred to as the provider. Providers and devices are connected by HTTP as shown in the illustration.

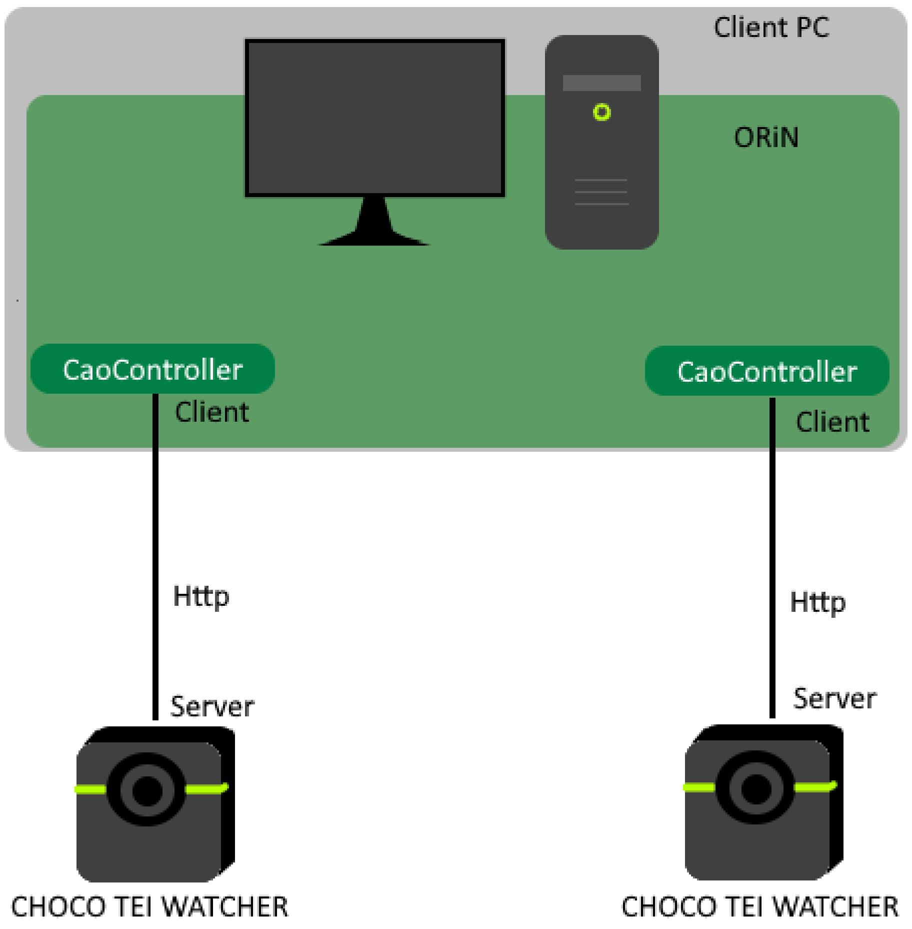


Fig. 1-1 Configuration Diagram

Fig. 1-2 shows the correspondence between this provider and each device.

(※ An example. It does not represent everything.)

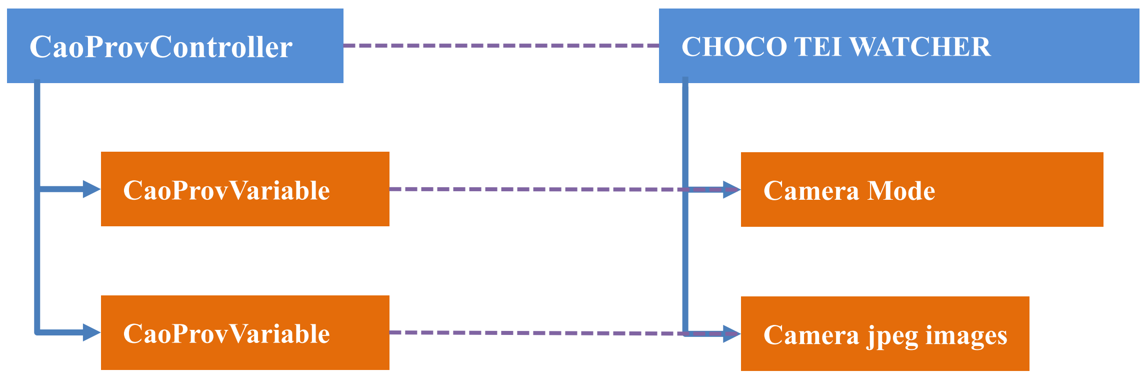


Fig. 1-2 Provider configuration and device information

2. Setting Up Your Environment for Application Development

2.1. Connecting Devices to a Client PC

This section describes the procedure for connecting a client PC to a device.

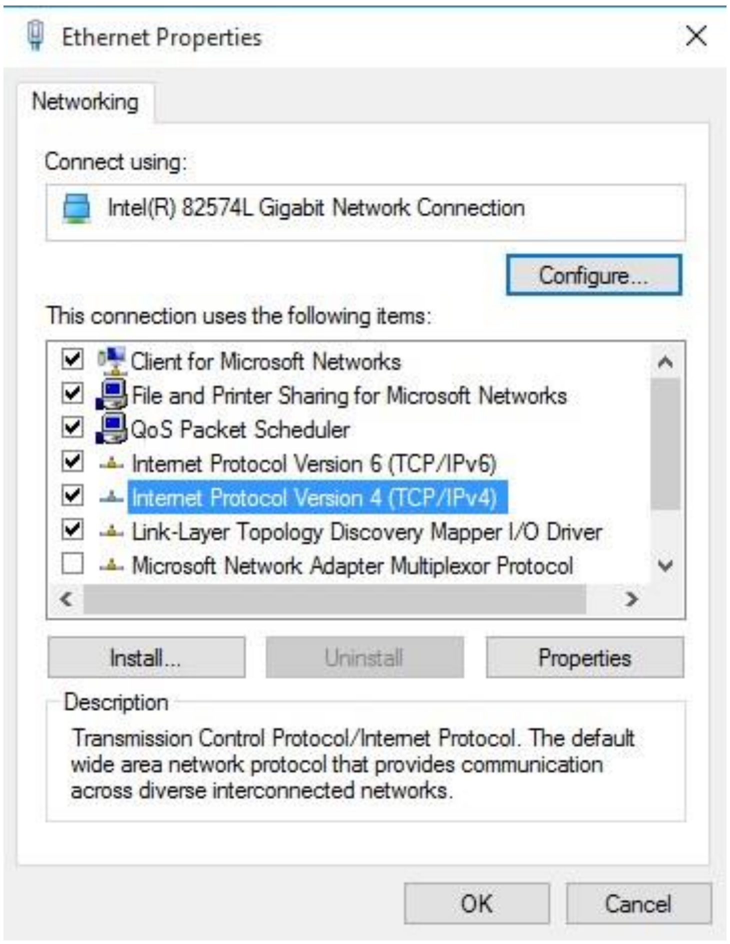
2.1.1. PC network settings

The client PC and the device are connected by a fixed IP address.

① Start the client PC, execute the "Start" → "Settings" → "Network and Internet" "Ethernet" → "Change adapter options" menu, and display the "Network Connection Settings" window.

② Open the Properties window for the wired LAN adapter that you want to connect to the device.

③ With the "Internet Protocol Version 4 (TCP/IPv4)" option selected in the Properties window, click the "Properties" button.


Ethernet Properties

Networking

Connect using:

Intel(R) 82574L Gigabit Network Connection

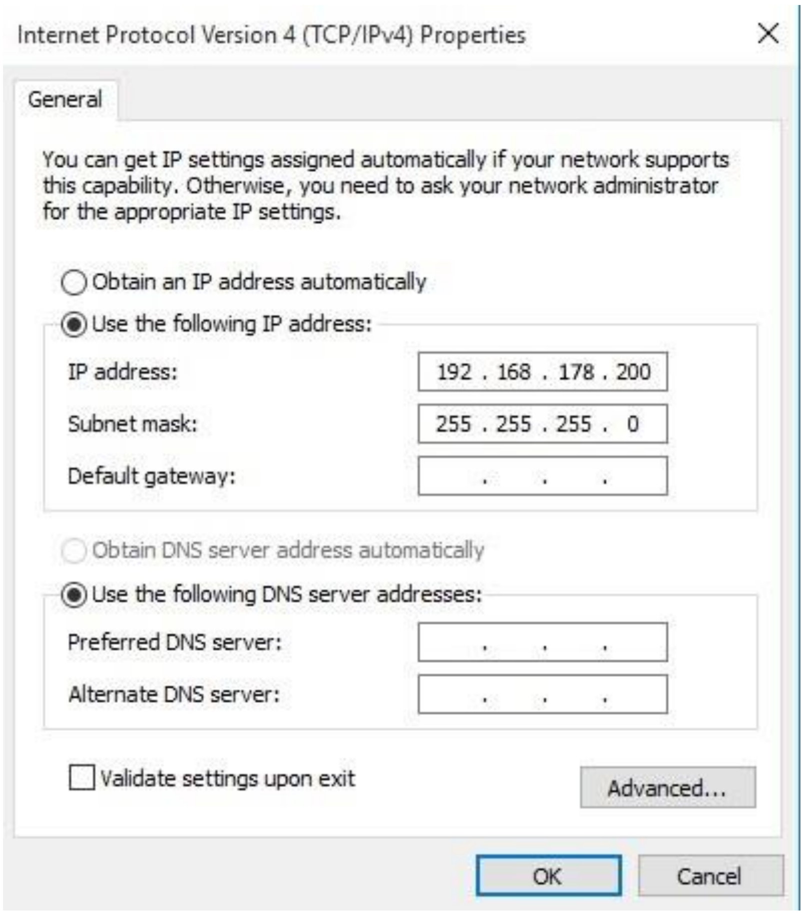
This connection uses the following items:

- [✓] Client for Microsoft Networks
- [✓] File and Printer Sharing for Microsoft Networks
- [✓] QoS Packet Scheduler
- [✓] Internet Protocol Version 6 (TCP/IPv6)
- [✓] Internet Protocol Version 4 (TCP/IPv4)
- [✓] Link-Layer Topology Discovery Mapper I/O Driver
- [ ] Microsoft Network Adapter Multiplexor Protocol

Description

Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

④ In IP Address and Subnet Mask, enter a value that can communicate with the device. The image shows an example of the IP address and subnet mask set on the client PC when the device IP address is 192.168.178.178 and the subnet mask is 255.255.255.0.


Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically
● Use the following IP address:

IP address: 192 . 168 . 178 . 200
Subnet mask: 255 . 255 . 255 . 0
Default gateway: . . .

○ Obtain DNS server address automatically
● Use the following DNS server addresses:

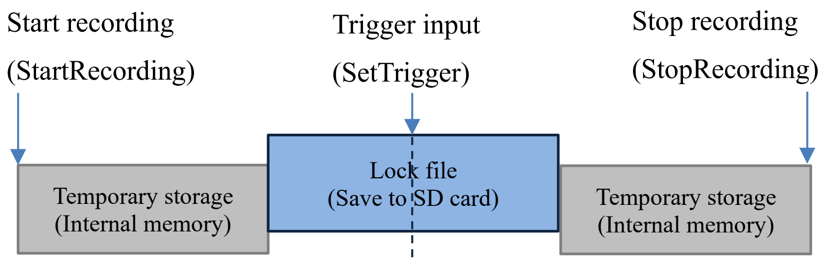
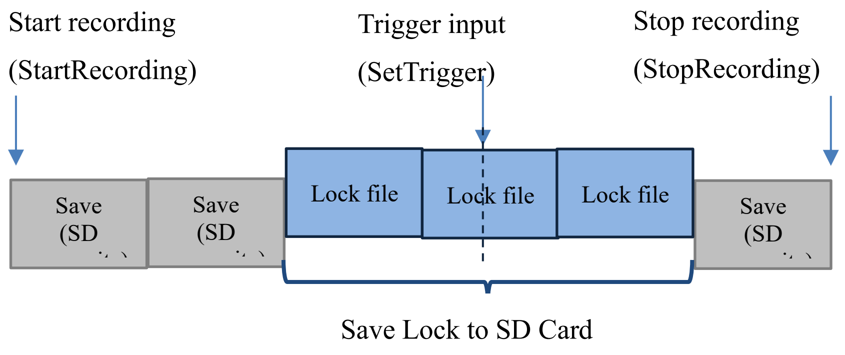
Preferred DNS server: . . .
Alternate DNS server: . . .

☐ Validate settings upon exit

2.1.2. Device Recording Mode Settings

There are two recording modes for the device: trigger only mode (DRAM mode) and loop and trigger lock mode (SD mode). Set the recording mode to suit your needs.

Record mode

Trigger Only Mode	A recording mode in which videos are temporarily stored in memory. When a trigger input is received, a movie file centered on trigger timing is saved on the SD card. Since access to the SD card is saved only by trigger input, the service life of the SD card can be extended. The recording unit can be selected from 20s and 40s. The figure below shows the behavior image in trigger-only mode and the correspondence between the provider's extended commands to perform each action.  Start recording (StartRecording) Trigger input (SetTrigger) Stop recording (StopRecording) Temporary storage (Internal memory) Lock file (Save to SD card) Temporary storage (Internal memory)
Loop and Trigger Lock Mode	Videos are stored repeatedly on the SD card for each recording unit. When a trigger input is received, the trigger timed movie and all three movies before and after it are saved with overwrite prohibited (locked) for three files. You can select the recording units from 1m, 3m, 5m and 10m. The figure below shows the behavior image when in Loop Trigger Lock mode and the correspondence between the provider's extended commands to perform each action.  Start recording (StartRecording) Trigger input (SetTrigger) Stop recording (StopRecording) Save (SD Save (SD Lock file Lock file Lock file Save (SD Save Lock to SD Card

Files that have been saved by accepting trigger input are referred to as lock files in the future.

2.1.3. Installing the Provider

Once you have installed ORiN2 SDKs, you are ready to use the providers.

3. Command Reference

3.1. Method/Property List

Table 3-1 List of methods and properties

Category	Methods/Properties[1]		Function	See Also
CaoWorkspace				
	AddController	M	Connected to controller	P.13
CaoController				
	GetVariableNames	M	Get a list of variable names that can be connected	P.14
	Variables	P	Get the variable collection held by the controller	P.14
	AddVariable	M	Add variable object	P.15
	Execute	M	Execute Extended Commands	P.15
	OnMessage	E	Message reception event	P.15
CaoVariable				
	Value	P	Get/set value	P.15

3.2. Method properties

3.2.1. CaoWorkspace classes

3.2.1.1. AddController method

Adds a controller object to CaoWorkspace. The provider connects to the device using the parameters passed when this method is executed. In addition, the @MODE variable-value acquisition process is executed to confirm the connection.

SYNOPSIS

```
AddController
(
        "<controller name>",            // Controller name (optional)
        "CaoProv.inaba.IB-MCT001",      // Provider name (fixed)
        "<machine name>",               // Provider execution machine name (unused)
        "<Option>"                      // Option character string
)
```

Option

The following is an optional specification for Option character string: Option character string is a comma (,)

[1] M: Indicates methods, P: properties, and E: events, respectively.

string consisting of the options listed below.

Option	Required	Description	Value Range	Default Value
Url	✓	Specify the URL to connect to. Normally, it is the following. Http://"Device-IP-Address"	-	-
Timeout	--	Specify the communication timeout (ms).	1 -	2000
BasePath	--	Specify the reference path to save the video. If a path relative to the destination path is specified when executing the movie acquisition command (GetMovie), the path relative to the reference path specified here will be used. If omitted, this is the directory where the provider dll is located.	--	Directory of the provider dll

Example (C#)

```
// Engine
ORiN2.ManagedCAO.CCaoEngine engine = new ORiN2.ManagedCAO.CCaoEngine();
// Workspace
ORiN2.ManagedCAO.CCaoWorkspace workspace = engine.AddWorkspace("NewWrks", "");
// Controller
ORiN2.ManagedCAO.CCaoController controller
    = workspace.AddController("IB-MCT001",
                                        "CaoProv.inaba.IB-MCT001",
                                        "",
                                         "url = http://192.168.178.178, timeout = 5000, basePath=C:¥¥");
```

3.2.2. CaoController classes

3.2.2.1. GetVariableNames method

Gets a list of variable names that can be connected.

SYNOPSIS

```
GetVariableNames
(
        "<Option>"                    // Option character string
)
```

3.2.2.2. Variables Properties

Gets a collection of variables that the controller holds.

Example (C#)

```
// Variable Collection Retrieval
ORiN2.ManagedCAO.CCaoVariables variables = controller.Variables;
```

```
// Variable acquisition
ORiN2.ManagedCAO.CCaoVariable variable = variables[0];
```

3.2.2.3. AddVariable method

Adds a variable object to CaoController. Only the variable names shown in 3.3.1 can be used.

AddVariable is specifi ed as follows.

SYNOPSIS

```
AddVariable
(
        "<variable name>",          // Variable name
        "<Option>"                  // Option character string (not used)
)
```

3.2.2.4. Execute method

Execute CaoController extension. Execute is specified as follows. An extended command list is defined in 3.4.

SYNOPSIS

```
Execute
(
        "<extension command name>",      // Extended command name
        "<Option string>"                // Option character string (optional)
)
```

3.2.2.5. OnMessage event

You can receive controller error notifications and status changes as OnMessage events. See 3.5 for the events that can be received.

3.2.3. CaoVariable classes

3.2.3.1. Value Properties

Acquires/sets data from the connected device. The behavior depends on the variable name. For details, refer to section 3.3, Variable List.

3.3. Variable list

Defines a list of variables that can be used in each class. Variables refer to objects of CaoVariable classes.

3.3.1. CaoController class-variable

Variable name	Description	Value		See Also
		Get	Put	
@MAKER_NAME	Obtain the manufacturer's name.	✓	-	P.16
@VERSION	Get the DLL version.	✓	-	P.16
@MODE	Gets/sets the camera mode. 0: Shooting mode 1: Playback mode	✓	✓	P.17
@CAMERA_INFO	Gets/sets the camera information.	✓	✓	P.18
@FILE_LIST	Gets the list of saved files.	✓	-	P.19
@IMAGE	Gets the present-camera images with JPEG codes.	✓	-	P.20
@CAPACITY	Used to acquire the remaining storage capacity of the camera.	✓	-	P.21
@ERROR_CODE	Gets the operating status and error code.	✓	-	P.21

3.3.1.1. @MAKER_NAME

Obtain the manufacturer's name.

Data Type

Type Description	
VT_BSTR	Obtain the manufacturer's name.

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@MAKER_NAME","");
// Acquisition of Values
string value = var.Value as string;
```

3.3.1.2. @VERSION

Gets the DLL version.

Data Type

Type Description	
VT_BSTR	Get the DLL version. *.*.*

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@VERSION","");
// Acquisition of Values
```

```
string value = var.Value as string;
```

3.3.1.3. @MODE

Gets/sets the camera mode. Some variable/extended commands have restrictions on the camera mode that can be executed. Before executing a variable or extended command that has restrictions on the camera mode, use this variable to set the camera mode, and then execute the command.

Execution availability in camera mode

The following table shows whether each variable can be executed in camera mode.

Variable		Shooting mode	Playback mode
@CAMERA_INFO	Get	✓	✓
	Put	-	✓
@FILE_LIST	Get	-	✓
@IMAGE	Get	✓	✓
@CAPACITY	Get	✓	✓
@ERROR_CODE	Get	✓	✓
@MAKER_NAME	Get	✓	✓
@VERSION	Get	✓	✓

The following shows whether or not each extended command can be executed in camera mode.

Extended Commands	Shooting mode	Playback mode
StartRecording	✓	-
StopRecording	✓	-
SetTrigger	✓	-
GetErrorLog	-	✓
GetFileList	-	✓
GetMovie	-	✓
DeleteFile	-	✓

Data Type

Type Description	
VT_I4	Camera Mode 0: Shooting mode 1: Playback mode

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@MODE","");
// Acquisition of Values
Int32? value = var.Value as Int32?;
// Set to playback mode
var.Value = 1;
```

3.3.1.4. @CAMERA_INFO

Gets/sets the camera information. The value of this variable can be set only when the camera mode is in playback mode. Also note that strings are case-sensitive.

Data Type

Type Description		
VT_VARIANT | VT_ARRAY		
0	VT_BSTR	Name (ASCII) Example). CAMERA
1	VT_DATE	Current time Example). 2020/10/20 12:15:32
2	VT_BSTR	String indicating the language JP: Japanese ENG: English
3	VT_BSTR	Text indicating the volume OFF: OFF S: Small M: Medium L: Large
4	VT_BSTR	Camera's recording mode For more information on recording modes, see 0.0 SD: Loop and trigger lock mode DRAM: Trigger Only Mode
5	VT_I4	Recording Time in Loop and Trigger Lock Mode (s) Value Range: 60, 180, 300, 600
6	VT_I4	Recording Time in Trigger Only Mode (s) Value Range: 20, 40
7	VT_I4	Camera Angle of View Setting (Degrees) Value Range: 110, 180
8	VT_BSTR	Version information Example). 72n

9	VT_I4	Factory Default Flag 0: Factory default 1: User-set status
10	VT_I4	Time to disconnect communication (min) If there is no operation, the connection to the camera is disconnected. 0 indicates unlimited. Value Range: 0, 3, 5, 10

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable cameraInfoVar = controller.AddVariable("@CAMERA_INFO","");
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");
// Get value
object[] cameraInfo = cameraInfoVar.Value as object[];

// Set CAMERA to Name and DRAM to Operating Mode
bool changeMode = false;
if ((modeVar.Value as Int32?).Value != 1)
{
        // When the camera mode is the shooting mode, set to the playback mode once.
        changeMode = true;
        modeVar.Value = 1;
}

cameraInfo[0] = "CAMERA";
cameraInfo[4] = "DRAM";

cameraInfoVar.Value = cameraInfo;

if (changeMode)
{
    modeVar.Value = 0;
}
```

3.3.1.5. @FILE_LIST

Gets the list of files saved in the camera. This variable can only be executed when the camera mode is in playback mode.

Data Type

Type Description	
VT_VARIANT | VT_ARRAY	
I	VT_VARINAT | VT_ARRAY

	0	VT_BSTR	The file path. The file path obtained here is used by GetMovie command and DeleteMovie command. Files other than lock files and lock files are determined by their name. **Lock file** "IP address suffix + alphanumeric (L or A) + three-digit index number.MOV" **Other than the lock file** "IP address suffix_3-digit index number.MOV"
	1	VT_DATE	Movie start date and time Example). 2020/10/28 09:00:00
	2	VT_DATE	Movie end date and time Example). 2002/10/28 09:01:00

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable fileListInfoVar = controller.AddVariable("@FILE_LIST","");
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

bool changeMode = false;
if ((modeVar.Value as Int32?).Value != 1)
{
    // When the camera mode is the shooting mode, set to the playback mode once.
    changeMode = true;
    modeVar.Value = 1;
}

object[] fileList = fileListInfoVar.Value as object[];

foreach (object file in fileList)
{
    // Some kind of processing
}

if (changeMode)
{
    modeVar.Value = 0;
}
```

3.3.1.6. @IMAGE

Gets the present-camera images with JPEG codes.

Data Type

Type Description

VT_UI1 | VT_ARRAY	Camera's JPEG graphics code. Immediately after switching the camera mode, the data may not exist.

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable imageVar = controller.AddVariable("@IMAGE","");
Byte[] jpegCodes = imageVar.Value as Byte[];
```

3.3.1.7. @CAPACITY

Used to acquire the remaining storage capacity of the camera.

Data Type

Type Description		
VT_VARIANT | VT_ARRAY		
0	VT_I4	Remaining storage capacity (sec)
1	VT_I4	Lock file availability 0: No lock file 1: With lock file
2	VT_DATE	Current date and time Example). 2020/10/28 10:10:15

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable capacityVar = controller.AddVariable("@CAPACITY","");
object[] capacities = capacityVar.Value as object[];

// Remaining storage capacity
Int32? capacity = capacities[0] as Int32?;
// Lock file availability
Int32? lockFile = capacities[1] as Int32?;
```

3.3.1.8. @ERROR_CODE

Gets the operating status and error code.

Data Type

Type Description
VT_VARIANT | VT_ARRAY

0		VT_BSTR	String indicating the operating status Use this to decide to execute StartRecording, StopRecording, SetTrigger command described later. The "T000" status is the default status of the device. Note that the device automatically enters the "T000" status after a certain period of inactivity even if the device has transitioned to another status. T000: No trigger during recording T001: Recording trigger exists T002: During playback T003: Recording stopped
1		VT_BSTR | VT_ARRAY	
	I	VT_BSTR	String indicating the error condition E000: Normal operation in progress E001: SD-Card Not Inserted E002: SD-Card not recognized E101: Ethernet initialization error E102: System error E103: File system error

Example (C#)

```
// Add Variable
ORiN2.ManagedCAO.CCaoVariable errorCodeVar = controller.AddVariable("@ERROR_CODE","");
object[] errorCodes = errorCodeVar.Value as object[];

// Operating state
string capacity = errorCodes [0] as string;
// Error status
string[] errorStates = errorCodes[1] as string[];
```

3.4. Extended command list

Describes the extended command list that can be used in Execute method of the controller class.

Command	Description	See Also
StartRecording	Recording starts. This can be performed only when the camera mode is set to the shooting mode (0).	P.24
StopRrecording	Recording stops. This can be performed only when the camera mode is set to the shooting mode (0).	P.25
SetTrigger	Sets a trigger and locks the video file being recorded. This can be performed only when the camera mode is set to the shooting mode (0).	P.25
GetErrorLog	Get the error log.	P.25
GetFileList	Gets the list of existing files. This function is available only when the camera mode is set to playback mode (1).	P.26
GetMovie	Acquires the specified video data asynchronously. This function is available only when the camera mode is set to playback mode (1).	P.27
DeleteFile	Removes the given file This function is available only when the camera mode is set to playback mode (1).	P.28

3.4.1. Creating a Lock File and Obtaining Video Files

This section describes the procedure for using the provider to generate a lock file and acquire the generated video file.

1. You switch to playback mode.
   Use @MODE to set the camera's mode to playback mode to get the current file list. (Set Value to 1.)
2. Get the current file list.
   Retrieves the value of the **@FILE_LIST** variable and stores the current file list in memory.
3. Change to shooting mode.
   To begin recording, set **@MODE** to 0 and change the camera's mode to capture mode.
4. You start recording.
   Execute **StartRecording** command. Recording starts.
5. Set the trigger.
   Execute SetTrigger command when you want to generate a lock file and start saving the lock file.
6. Wait for the lock file to finish saving.
   Gets the **@ERROR_CODE** and waits until the operational status is no longer triggered ("T001"). We recommend that you do this asynchronously.
7. You switch to playback mode.
   When the locked file is saved, use **@MODE** to change to Play mode to retrieve the video file.

8. Get the current file list.

   Use the **@FILE_LIST** variable to obtain a list of files after saving the lock file.

9. Extract the newly created lock file compared to the file list obtained in step 2.

   Compare the old and new file lists to get the file path of the newly created lock file. Use this file name in the next step.

10. Obtain a movie file.

    Execute **GetMovie** command. Save the generated lock file.

3.4.2. StartRecording

Recording starts. Perform the camera mode to a recording mode. The device defaults to the record start state.

Data Type

Item	Type Description
Argument	None
Return Value	None

Example (C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");
ORiN2.ManagedCAO.CCaoVariable errorCodeVar = controller.AddVariable("@ERRO_CODE", "");
if ((modeVar.Value as Int32?).Value != 0)
{
    // When the camera mode is the playback mode, set to the shooting mode once.
    modeVar.Value = 0;
}

// Starting Recording (Normally this operation is not required)
controller.Execute("StartRecording", "");

// Lock file save start
controller.Execute("SetTrigger", "");

// Wait until the lock file is saved.
await System.Threading.Tasks.Task.Run(new Action(() =>
{
     while(true)
    {
        object[] errorCodeValue = errorCodeVar.Value as object[];
        string status = errorCodeValue[0] as string;
        if (status != "T001")
        {
            break;
        }
        System.Threading.Thread.Sleep(0);
    }
}));

// Stop recording (execute if necessary)
```

```
controller.Execute("StopRecording", "");

// Please refer to GetMovie if you want to acquire a movie later.
```

3.4.3. StopRecording

Recording ends. Perform the camera mode to a recording mode. The default state of the device is Record. We recommend that you do not execute this command unless you have a special reason for doing so.

Data Type

Item	Type Description
Argument	None
Return Value	None

Example (C#)

Refer to StartRecording.

3.4.4. SetTrigger

Set the trigger and create a lock file. Run while recording starts.

Data Type

Item	Type Description
Argument	None
Return Value	None

Example (C#)

Refer to StartRecording.

Lock file creation complete

While a lock file is being created, "T001" is displayed for the @ERROR_CODE operating status string. Determine whether the lock file has been created using the @ERROR_CODE operation status string.

3.4.5. GetErrorLog

Retrieves the error log held by the camera as a character string. Switch the camera mode to Play mode and execute it.

Data Type

Item	Type Description	
Argument	None	
Return Value	VT_BSTR | VT_ARRAY	
	I	Error Log

Example (C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

if ((modeVar.Value as Int32?).Value != 1)
{
    // If the camera mode is set to playback mode, set it to playback mode once.
    modeVar.Value = 1;
}

string[] errorLog = controller.Execute("GetErrorLog", "") as string[];
```

3.4.6. GetFileList

Gets the list of files saved in the camera. Switch the camera mode to Play mode and execute it.

Data Type

Item	Type Description			
Argument	None			
Return Value	VT_VARIANT | VT_ARRAY			
	I	VT_VARINAT | VT_ARRAY		
		0	VT_BSTR	The file path. The file path obtained here is used by GetMovie command and DeleteMovie command. Files other than lock files and lock files are determined by their name. **Lock file** "IP address suffix + alphanumeric (L or A) + three-digit index number.MOV" **Other than the lock file** "IP address suffix_3-digit index number.MOV"
		1	VT_DATE	Movie start date and time Example). 2020/10/28 09:00:00
		2	VT_DATE	Movie end date and time Example). 2002/10/28 09:01:00

Example (C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

if ((modeVar.Value as Int32?).Value != 1)
{
    // If the camera mode is set to playback mode, set it to playback mode once.
     modeVar.Value = 1;
}

object[] fileList = controller.Execute("GetFileList", "") as object[];

foreach (object file in fileList)
{
    // Some kind of processing
}
```

3.4.7. GetMovie

Saves the video file specified by the argument to the specified path. This command is executed asynchronously, and the completion of the execution is notified by OnMessage. Note, however, that you cannot execute multiple operations at the same time. Also, set the camera mode to playback mode.

Data Type

Item	Type Description		
Argument	VT_BSTR | VT_ARRAY		
	0	VT_BSTR	Specifies the file path of the destination.
	1	VT_BSTR	Specify the destination path. You can specify a path either relative or absolute. If a relative path is specified, the path is relative to the reference path specified during AddController. If not specified, the reference path is the path where the provider dll is located. If a directory does not exist in the path, it is created automatically, but if the path requires administrator privileges, the directory creation fails. In this case, please create a directory beforehand. **If you do not specify file names,** If you specify a file without including the file name, the following file is saved. "Specified directory" + "Name of file to retrieve from" If you want to specify this, be sure to specify "/" or "¥" at the end. If there are no such characters at the end, the file name is assumed. **To specify up to the file name** If you specify a file name, the file is saved to the specified file path. Note that existing files will be overwritten.
Return Value	None		

Example (C#)

```
// Engine
private ORiN2.ManagedCAO.CCaoEngine engine;
```

```
// Workspace
private ORiN2.ManagedCAO.CCaoWorkspace workspace;
// Controller
private ORiN2.ManagedCAO.CCaoController controller;

///<summary>
/// Sample code for GetMovie
///</summary>
public void GetMovieSample()
{
        // Engine
        this.engine = new ORiN2.ManagedCAO.CCaoEngine();
        // Workspace
        this.workspace = engine.AddWorkspace("NewWrks", "");
        // Controller
        this.controller = this.workspace.AddController("IB-MCT001",
                                              "CaoProv.inaba.IB-MCT001",
                                              "",
                                              "url = http://192.168.178.178, timeout = 5000");
        this.controller.OnMessage += this.OnMessage;

        this.controller.Execute("GetMovie", new string[2] {@"Sample/Sample.mov",
                                                              @"C://Sample/Sample.mov"});
}


///<summary>
/// OnMessage event handlers
///</summary>
private void OnMessage (object sender, OnMessageEventArgs e)
{
      if (e.Message.Number == 0)
      {
           // Success
      }
}
```

3.4.8. DeleteFile

Deletes the file in the camera specified by the argument.

Data Type

Item	Type Description	
Argument	VT_BSTR	Specify the file path to delete.
Return Value	None	

Example (C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

if ((modeVar.Value as Int32?).Value != 1)
{
    // If the camera mode is set to playback mode, set it to playback mode once.
    modeVar.Value = 1;
}

controller.Execute("DeleteFile", "DeleteFilePath");
```

3.5. Event list

GetMovie extension command executed asynchronously can be received by OnMessage event.

No.	Description
0	Video acquisition completion event This event is issued when the video acquisition is completed successfully.
1	File operation failure event An event that is issued when a file operation fails.
2	Movie data empty event This event is issued when the acquired video data is empty.

Value Property Data Types

The data type is common for all events.

Type Description		
VT_BSTR | VT_ARRAY		
0	VT_BSTR	Path to the downloaded movie file
1	VT_BSTR	Full path to the destination where the downloaded movie data is saved Make sure that the path specified in this property is correct when a file operation failure event is received.

4. PROGRAMMING BY PROVIDERS

The providers allow you to connect the client PC to CHOCO TEI WATCHER as follows:

- Creating a CaoEngine
- Creating a CaoWorkspace
- Creating a CaoController

After you connect to CHOCO TEI WATCHER, you can access CaoController by using its Execute method or by creating a CaoVariable object.

4.1. Sample programming to acquire the current image

In this example, we will create a sample program that gets the present picture of CHOCO TEI WATCHER. Table 4-1 describes the requirements of the sample program, and Fig. 4-1 describes the flow of the sample program.

Table 4-1 Sample program requirements

Requirements	Description
Host	The destination IP address is 192.168.178.178
Process Description	Get the present picture of CHOCO TEI WATCHER.

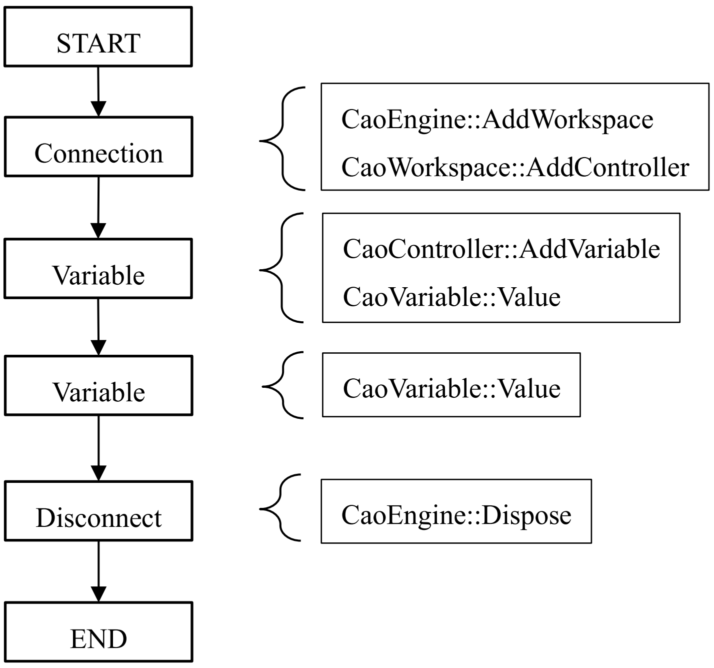


Fig. 4-1 Process flow

Specific codes are given in the following sections.

4.1.1. Sample program

The following is an overview of the sample program.

Sample	GetImageSample.cs

```
// Object
private ORiN2.ManagedCAO.CCaoEngine m_caoEngine = null;
private ORiN2.ManagedCAO.CCaoWorkspace m_caoWorkspace = null;
private ORiN2.ManagedCAO.CCaoController m_caoController = null;
private ORiN2.ManagedCAO.CCaoVariable m_varImage = null;
public void Main()
{
        // Connection
        this.Connect();
        // Adding Variables and Retrieving Values
        this.m_varImage = this.m_caoController.AddVariable("@IMAGE","");
        Byte[] jpegCodes = this.m_varImage.Value as Byte[];

        // Disconnect
        this.Disconnect();
}

// Connection method
private void Connect()
{
        // Generate CaoEngine object
        this.m_caoEngine = new ORiN2.ManagedCAO.CCaoEngine();
        // Generate CaoWorkspace object
        this.m_caoWorkspace = this.m_caoEngine.AddWorkspace("NewWrks", "");
        // Generate CaoController object
        this.m_caoController = this.m_caoWorkspace.AddController("IB-MCT001",
                                        "CaoProv.inaba.IB-MCT001",
                                        "",
                                        "url = http://192.168.178.178, timeout = 5000, basePath=C:¥¥");
}

// Disconnect method
private void Disconnect()
{
        this.m_caoEngine.Dispose();
        this.m_caoEngine = null;
}
```

4.1.1.1. Connection

To connect to CHOCO TEI WATCHER, proceed as follows:

(1) Prepare a variable to hold the object. The objects required to connect to the controller are CaoEngine object, CaoWorkspace object, and CaoController object. CaoWorkpace object does not need to have a variable to obtain CaoController object from CaoWorkspaces. You will also need a CaoVariable for accessing the variable. The following is a code example for C#.

```
        // Variables for CaoEngine
```

```
private ORiN2.ManagedCAO.CCaoEngine m_caoEngine = null;
// Variables for CaoWorkspace
private ORiN2.ManagedCAO.CCaoWorkspace m_caoWorkspace = null;
// Variables for CaoController
private ORiN2.ManagedCAO.CCaoController m_caoController = null;
// Variables for CaoVariable
private ORiN2.ManagedCAO.CCaoVariable m_varImage = null;
```

(2) Creates a CaoEngine object. CaoEngine object is generated using the New keyword.

```
// Generate CaoEngine object
this.m_caoEngine = new ORiN2.ManagedCAO.CCaoEngine();
```

(3) Gets or generates a CaoWorkspace object. When you create a CaoEngine object, it defaults to one CaoWorkspaces object and one object. The following is a sample code/default CaoWorkspace for creating a new CaoWorkspace.

```
// Generate CaoWorkspace object
this.m_caoWorkspace = this.m_caoEngine.AddWorkspace("NewWrks", "");
```

(4) Create a CaoController object. To generate a CaoController object, set the provider name to use and the parameters to use.

```
// Generate CaoController object
this.m_caoController = this.m_caoWorkspace.AddController("IB-MCT001",
                                    "CaoProv.inaba.IB-MCT001",
                                    "",
                                    "url = http://192.168.178.178, timeout = 5000, basePath=C:¥¥");
```

4.1.1.2. Adding Variables and Retrieving Values

Use the @IMAGE variable to retrieve the image data.

```
// Adding Variables and Retrieving Values
this.m_varImage = this.m_caoController.AddVariable("@IMAGE","");
Byte[] jpegCodes = this.m_varImage.Value as Byte[];
```

4.1.1.3. Disconnect

To disconnect from the controller, you can erase the generated objects and delete the objects that you want to erase from the collection class that manages the objects. However, you do not need to explicitly remove it if you use ORiN.ManagedCAO. The following is a code example:

```
// Remove all objects from CaoEngine
this.m_caoEngine.Dispose();
// Clear CaoEngine
this.m_caoEngine = null;
```

5. Provider error code

This provider has the following unique error codes masked by 0x8011xxxxx through 0x8015xxxxx: (Refer to Table 5-1 Unique Error Codes Table.)

Table 5-1 Unique Error Codes

Error Number	Description
0x80110001	Failed to specify the URL option during AddController. Be sure to specify URLs during AddController.
0x80110002	Timeout optional specification value error during AddController. Raised when a value outside the value range of Timeout option is set. Specify 1 to 2147483647 for Timeout.
0x80110003	BasePath optional specification value error during AddController. Occurs when the reference path specified by BaePath option is incorrect. Specify the correct reference path.
0x80111xxx	HTTP failure during AddController. Occurs when communication confirmation at the time of AddController fails. Check that the URL specified in the URL option is correct. The actual HTTP response code is stored in hexadecimal in the last three digits ("xxx").
0x80121xxx	HTTP failure when executing extended command. An HTTP failure occurred while executing the extended command. Check the connection status between the device and the PC. The actual HTTP response code is stored in hexadecimal in the last three digits ("xxx").
0x801220xx	Data error during extended command execution. The acquired data contains unexpected data, or the argument specified when the command was executed is incorrect. For xx, refer to the extended command number table.
0x801230xx	Command execution error Error from the camera.
0x80124006	Duplicate Video Acquisition Command Error An attempt was made to execute another video command acquisition command while a video acquisition command was being executed. Wait until the video acquisition command being executed ends.
0x801250xx	Extended command camera mode error. A command was executed in a camera mode that is not permitted.

Error Number	Description
	For xx, refer to the extended command number table.
0x80130000	Variable name error. An unsupported variable name was specified.
0x80141xxx	HTTP failure when acquiring data. A HTTP failure occurred while getting the data. The actual HTTP response code is stored in hexadecimal in the last three digits ("xxx").
0x801420xx	Acquisition data error. There was data that could not be converted when the value was acquired. For xx, refer to the variable number table.
0x801450xx	A value acquisition was executed in a camera mode that is not permitted. For xx, refer to the variable number table.
0x80151xxx	HTTP failure during setting. A HTTP failure occurred during setting. The actual HTTP response code is stored in hexadecimal in the last three digits ("xxx").
0x801520xx	Setting data error. The data specified in the value setting is incorrect. For xx, refer to the variable number table.
0x801530xx	Error from the camera when setting the value. An error response was returned from the camera when the value was set. For xx, refer to the camera error code table.
0x801540xx	Value setting unsupported error You are attempting to set a value for a variable that does not support value setting. For xx, refer to the variable number table.
0x801550xx	A value setting was executed in a camera mode that is not permitted. For xx, refer to the variable number table.

Extended command number table

Extended command number	Extended Commands
0x01	StartRecording
0x02	StopRecording
0x03	SetTrigger
0x04	GetErrorLog
0x05	GetFileList
0x06	GetMovie

Extended command number	Extended Commands
0x07	DeleteFile

Variable number table

Variable number	Variable
0x01	@MAKER_NAME
0x02	@VERSION
0x03	@MODE
0x04	@CAMERA_INFO
0x05	@FILE_LIST
0x06	@IMAGE
0x07	@CAPACITY
0x08	@ERROR_CODE

CAMERA ERROR CODE TABLE

Error code	Variable
0x01	Wrong mode
0xFF	Command execution failed (-1)
0xFE	Trigger number error (-2)